It is claimed:

1. A method comprising:

receiving a PADD instruction comprising a result register identifier to identify a result register, a first source register identifier to identify a first source register storing a first operand, and an operand identifier to identify a second operand; and

in response to receiving said PADD instruction, causing a dedicated PADD logic device to perform a packet addition of the first and second operands to generate a result, and to subsequently store the result in said result register.

2. The method of claim 1, wherein said operand identifier identifies a second source register storing said second operand.

3. The method of claim 1, wherein said operand identifier identifies an immediate value to use as said second operand.

4. The method of claim 1, wherein said PADD instruction further comprises a start identifier to identify the start bit of said first operand.

5. The method of claim 1, wherein said PADD instruction further comprises a stop identifier to identify the stop bit of said first operand.

6. The method of claim 1, wherein said PADD instruction includes a carry in parameter to cause said dedicated PADD logic device to perform a packet addition of said first and second operands with a carry in to generate said result.

7. The method of claim 1, wherein said PADD instruction includes a data field parameter to indicate that only certain bits of the content stored in said first register serves as said first operand.

8. The method of claim 1, wherein said PADD instruction includes an immediate value parameter to indicate that an immediate value serves as said second operand.

9. An apparatus comprising:

a result register;

a first source register;

an instruction control device to receive a PADD instruction comprising a result register identifier to identify said result register, a first source register identifier to identify said first source register storing a first operand, and an operand identifier to identify a second operand; and

a dedicated PADD logic device responsive to said instruction control device to perform a packet addition of the first and second operands to generate a result subsequently stored in said result register.

10. The apparatus of claim 9, further comprising a second source register, wherein said operand identifier identifies said second source register as storing said second operand.

11. The apparatus of claim 9, wherein said operand identifier identifies an immediate value to use as said second operand.

12. The apparatus of claim 9, wherein said PADD instruction further comprises a start identifier to identify the start bit of said first operand.

13. The apparatus of claim 9, wherein said PADD instruction further comprises a stop identifier to identify the stop bit of said first operand.

14. The apparatus of claim 9, wherein said PADD instruction includes a carry in parameter to cause said dedicated PADD logic device to perform a packet addition of said first and second operands with a carry in to generate said result.

15. The apparatus of claim 9, wherein said PADD instruction includes a data field parameter to indicate that only certain bits of the content stored in said first register serves as said first operand.

16. The apparatus of claim 9, wherein said PADD instruction includes an immediate value parameter to indicate that an immediate value serves as said second operand.

17. A method comprising:

receiving a SMAD instruction comprising a result register identifier to identify a result register, a first source register identifier to identify a first source register storing a first operand, and a second source register identifier to identify a second source register storing a second operand; and

in response to receiving said SMAD instruction, causing a dedicated SMAD logic device to perform an addition of the first and second operands to generate a result, and to subsequently store the result in said result register.

18. The method of claim 17, wherein said SMAD instruction includes an accumulate parameter that causes the result generated by the SMAD logic device to be added to an existing content of said result register.

19. The method of claim 17, wherein said SMAD instruction further comprises a 2^n - 1 modulo parameter to cause said SMAD logic device to perform a 2^n - 1 modulo addition of said first and second operands.

20. The method of claim 17, wherein said SMAD instruction further comprises a length parameter to specify the bit length of said first and second operands.

21. The method of claim 17, wherein said SMAD instruction includes a number of operands parameter to specify the number of operands including the first and second operands which said SMAD logic device is to perform an addition to generate said result.

22. An apparatus comprising:

a result register;

a first source register;

a second source register;

an instruction control device to receive a SMAD instruction comprising a result register identifier to identify said result register, a first source register identifier to identify said first source register storing a first operand, and a second source register identifier to identify said second source register storing a second operand; and

a dedicated SMAD logic device responsive to said instruction control device to perform an addition of the first and second operands to generate a result which is subsequently stored in said result register.

23. The apparatus of claim 22, wherein said SMAD instruction includes an accumulate parameter that causes the result generated by the dedicated SMAD logic device to be added to an existing content of said result register.

24. The apparatus of claim 22, wherein said SMAD instruction further comprises a 2^n - 1 modulo parameter to cause said dedicated SMAD logic device to perform a 2^n - 1 modulo addition of said first and second operands.

25. The apparatus of claim 22, wherein said SMAD instruction further comprises a length parameter to specify the bit length of said first and second operands.

26. The apparatus of claim 22, wherein said SMAD instruction includes a number of operands parameter to specify the number of operands including the first and second operands which said dedicated SMAD logic device is to perform an addition to generate said result.

27. A method comprising:

receiving a first source packet comprising a first source data field situated at the least significant bits of said first source packet, a second source data field adjacent to and on a more significant bit side of said first source data field, and a third source data field adjacent to and on a more significant bit side of said second source data field;

receiving a second source packet comprising a fourth source data field situated at the least significant bits of said second source packet; and

forming a result packet comprising a first result data field being the same bitwise and bit position as said first source data field, a second result data field being the addition of said second and fourth source data fields in the same bit position as said second source data field, and a third result data field being the same bitwise and bit position as said third source data field.

28. The method of claim 27, wherein forming said result packet comprises:

forming a first intermediate packet comprising a first intermediate data field being all non-asserted bits in the same bit position and length as said first source data field, a second intermediate data field being the same bitwise as said fourth source data field and in the same bit position as said second source data field, and a third intermediate data field being all asserted bits in the same bit position and length as said third source data field;

forming a second intermediate packet comprising a fourth intermediate data field being the same bitwise and bit position as said first source data field, a fifth intermediate data field being the same bitwise and bit position as said second source data field, and a sixth intermediate data field being all non-asserted bits in the same bit position and length as said third source data field;

forming a third intermediate packet comprising a seventh intermediate data field being the same bitwise and bit position as said first source data field, and an eighth intermediate data field being the addition of said second and fifth intermediate data fields;

forming a fourth intermediate packet comprising a ninth intermediate data field being the same bitwise and bit position as said first source data field, a tenth intermediate data field being the same bitwise and bit position as said eighth intermediate data field, and an eleventh intermediate data field being all non-asserted bits at the same bit position as said third source data field;

forming a fifth intermediate packet comprising a twelfth data field comprising all non-asserted bits at the same bit position as said first and second source data fields, and a thirteen data field being the same bitwise and bit position as said third source data field; and

forming said result packet from said fourth and fifth intermediate packets.

29. The method of claim 28, wherein forming said third intermediate packet comprises adding said first and second intermediate packets.

30. The method of claim 28, wherein forming said result packet comprises bitwise ORing said fourth and fifth intermediate packets.

31. The method of claim 27, wherein forming said result packet comprises:

forming a first intermediate packet comprising a first intermediate data field being all asserted bits in the same bit position and length as said first source data field, a second intermediate data field being the same bitwise as said fourth source data field and in the same bit position as said second source data field, and a third intermediate data field being all asserted bits in the same bit position and length as said third source data field;

forming a second intermediate packet comprising a fourth intermediate data field being the same bitwise and bit position as said first source data field, a fifth intermediate data field being the same bitwise and bit position as said second source data field, and a sixth intermediate data field being all non-asserted bits in the same bit position and length as said third source data field;

forming a third intermediate packet comprising a seventh intermediate data field being the same bitwise and bit position as said first source data field, and an eighth intermediate data field being the addition of said second and fifth intermediate data fields;

forming a fourth intermediate packet comprising a ninth intermediate data field being the same bitwise and bit position as said first source data field, a tenth intermediate data field being the same bitwise and bit position as said eighth intermediate data field, and an eleventh intermediate data field being all non-asserted bits at the same bit position as said third source data field;

forming a fifth intermediate packet comprising a twelfth data field comprising all non-asserted bits at the same bit position as said first and second source data fields, and a thirteen data field being the same bitwise and bit position as said third source data field; and

forming said result packet from said fourth and fifth intermediate packets.

32. The method of claim 31, wherein forming said third intermediate packet comprises adding said first and second intermediate packets with a carry in.

33. The method of claim 31, wherein forming said result packet comprises bitwise ORing said fourth and fifth intermediate packets.

34. An apparatus comprising:

a logic device to:

receive a first source packet comprising a first source data field situated at the least significant bits of said first source packet, a second source data field adjacent to and on a more significant bit side of said first source data field, and a third source data field adjacent to and on a more significant bit side of said second source data field;

receive a second source packet comprising a fourth source data field situated at the least significant bits of said second source packet; and

form a result packet comprising a first result data field being the same bitwise and bit position as said first source data field, a second result data field being the addition of said second and fourth source data fields in the same bit position as said second source data field, and a third result data field being the same bitwise and bit position as said third source data field.

35. The apparatus of claim 34, wherein said logic device comprises:

a first sub-logic device to form a first intermediate packet comprising a first intermediate data field being all non-asserted bits in the same bit position and length as said first source data field, a second intermediate data field being the same bitwise as said fourth source data field and in the same bit position as said second source data field, and a third intermediate data field being all asserted bits in the same bit position and length as said third source data field;

a second sub-logic device to form a second intermediate packet comprising a fourth intermediate data field being the same bitwise and bit position as said first source data field, a fifth intermediate data field being the same bitwise and bit position as said second source data field, and a sixth intermediate data field being all non-asserted bits in the same bit position and length as said third source data field;

a third sub-logic device to form a third intermediate packet comprising a seventh intermediate data field being the same bitwise and bit position as said first source data field, and an eighth intermediate data field being the addition of said second and fifth intermediate data fields;

a fourth sub-logic device to form a fourth intermediate packet comprising a ninth intermediate data field being the same bitwise and bit position as said first source data field, a tenth intermediate data field being the same bitwise and bit position as said eighth intermediate data field, and an eleventh intermediate data field being all non-asserted bits at the same bit position as said third source data field;

a fifth sub-logic device to form a fifth intermediate packet comprising a twelfth data field comprising all non-asserted bits at the same bit position as said first and second source data fields, and a thirteen data field being the same bitwise and bit position as said third source data field; and

a sixth sub-logic device to form said result packet from said fourth and fifth intermediate packets.

36. The apparatus of claim 35, wherein said first sub-logic device comprises:

a left shifter to left shift said fourth source data field of said second source packet to the same bit position as said second source data field with leading logic zeros; and

a bitwise OR logic device to bitwise OR said shifted source packet with a mask to form said first intermediate packet.

37. The apparatus of claim 35, wherein said second sub-logic device comprises a bitwise AND logic device to bitwise AND said first source packet with a mask to form said second intermediate packet.

38. The apparatus of claim 35, wherein said third sub-logic device comprises an adder to add said first intermediate packet to said second intermediate packet to form said third intermediate packet.

39. The apparatus of claim 35, wherein said fourth sub-logic device comprises a bitwise AND logic device to bitwise AND said third intermediate packet with a mask to form said fourth intermediate packet.

40. The apparatus of claim 35, wherein said fifth sub-logic device comprises a bitwise AND to bitwise AND said first source packet with a mask to form said fifth intermediate packet.

41. The apparatus of claim 35, wherein said sixth sub-logic device comprises a bitwise OR logic device to bitwise OR said fourth and fifth intermediate packets to form said result packet.

42. The apparatus of claim 34, wherein forming said result packet comprises:

a first sub-logic device to form a first intermediate packet comprising a first intermediate data field being all asserted bits in the same bit position and length as said first source data field, a second intermediate data field being the same bitwise as said fourth source data field and in the same bit position as said second source data field, and a third intermediate data field being all asserted bits in the same bit position and length as said third source data field;

a second sub-logic device to form a second intermediate packet comprising a fourth intermediate data field being the same bitwise and bit position as said first source data field, a fifth intermediate data field being the same bitwise and bit position as said second source data field, and a sixth intermediate data field being all non-asserted bits in the same bit position and length as said third source data field;

a third sub-logic device to form a third intermediate packet comprising a seventh intermediate data field being the same bitwise and bit position as said first source data field,

and an eighth intermediate data field being the addition of said second and fifth intermediate data fields;

a fourth sub-logic device to form a fourth intermediate packet comprising a ninth intermediate data field being the same bitwise and bit position as said first source data field, a tenth intermediate data field being the same bitwise and bit position as said eighth intermediate data field, and an eleventh intermediate data field being all non-asserted bits at the same bit position as said third source data field;

a fifth sub-logic device to form a fifth intermediate packet comprising a twelfth data field comprising all non-asserted bits at the same bit position as said first and second source data fields, and a thirteen data field being the same bitwise and bit position as said third source data field; and

a sixth sub-logic device to form said result packet from said fourth and fifth intermediate packets.

43. The apparatus of claim 42, wherein said third sub-logic device comprises an adder to add said first intermediate packet to said second intermediate packet with a carry in to form said third intermediate packet.

44. A method, comprising:

generating a carry and a save from a plurality of operands;

adding the carry and the save together without a carry in to form a first sum;

adding the carry and the save together with a carry in to form a second sum;

selecting said first sum if a modulo 2^n addition is desired;

selecting said first sum if a modulo 2^n -1 addition is desired and said first adding does not generate a carry out; and

selecting said second sum if said modulo 2^n -1 addition is desired and said first adding does generate a carry out.

45. The method of claim 44, wherein generating said carry and said save is performed by a carry-save adder.

46. The method of claim 44, wherein generating said carry and said save comprises:

generating first and second intermediate carries;

generating first and second intermediate saves; and

generating said carry and said save from said first and second intermediate carries and said first and second intermediate saves.

47. The method of claim 46, wherein generating said first and second intermediate carries is performed by a first level carry-save adder, and wherein generating said carry and said save is performed by a second level carry-save adder.

48. The method of claim 44, wherein generating said carry and said save comprises:

generating first level carries and first level saves;

generating second level carries and second level saves from said first level carries and first level saves; and

generating said carry and said save from said second level carries and second level saves.

49. The method of claim 48, wherein generating said first level carries and first level saves are performed by a first level carry-save adder, wherein generating said second level carries and second level saves are performed by a second level carry-save adder, and wherein generating said carry and said save is performed by a third level carry-save adder.

50. The method of claim 44, wherein adding the carry and the save together to form a first sum and adding the carry and the save together with a carry in to form a second sum are performed substantially in parallel.

51. The method of claim 44, wherein selecting said first sum or said second sum is performed by a multiplexer.

52. An apparatus, comprising:

a carry-save adder to generate a carry and a save from a plurality of operands;

a first adder to generate a first sum of said carry and said save, wherein said first adder has a non-asserted carry in;

a second adder to generate a second sum of said carry and said save, wherein said second adder has an asserted carry in; and

a multiplexer to select as its output said first sum or said second sum.

53. The apparatus of claim 52, wherein said carry-save adder comprises:

a first level carry-save adder to generate first and second intermediate carries and first and second intermediate saves from said operands; and

a second level carry-save adder to generate said carry and said save from said first and second intermediate carries and said first and second intermediate saves.

54. The apparatus of claim 52, wherein said carry-save adder comprises:

a first level carry-save adder to generate first level carries and first level saves from said operands;

a second level carry-save adder to generate second level carries and second level saves from said first level carries and said first level saves; and

a third level carry-save adder to generate said carry and save from said second level carries and said second level saves.